11. Write a program that declares an abstract class Shape that implements the Drawable method and also includes a field String color and an abstract method calculateArea(). Implements two subclasses: Circle → stores radius, overrides both methods Rectangle → stores length and breadth, overrides both methods In the main() method: Use polymorphism to refer to Shape references and call overridden methods

Code

```
abstract class Shape {
    String color;

    Shape(String color) {
        this.color = color;
    }

    // Abstract methods
    abstract void draw();
    abstract double calculateArea();
}

// Circle subclass
class Circle extends Shape {
    double radius;
```

```
    Circle(String color, double radius) {
        super(color);
        this.radius = radius;
    }

    void draw() {
        System.out.println("Drawing Circle of color " + color);
    }

    double calculateArea() {
        return 3.14 * radius * radius;
    }
}

// Rectangle subclass
class Rectangle extends Shape {
    double length, breadth;

    Rectangle(String color, double length, double breadth) {
        super(color);
        this.length = length;
        this.breadth = breadth;
    }

    void draw() {
```

```
        System.out.println("Drawing Rectangle of color " +
color);
    }

    double calculateArea() {
        return length * breadth;
    }
}

public class AbstractDemo {
    public static void main(String[] args) {

        // Polymorphism
        Shape s1 = new Circle("Red", 5);
        Shape s2 = new Rectangle("Blue", 4, 6);

        s1.draw();
        System.out.println("Area of Circle: " +
s1.calculateArea());

        System.out.println();

        s2.draw();
        System.out.println("Area of Rectangle: " +
s2.calculateArea());
    }
```

```
}
```